N89-20696

OFMTutor

An Operator Function Model Intelligent Tutoring System

Patricia M. Jones

ISYE 8706

8 November 1988

INTRODUCTION

This paper proposes the design, implementation, and evaluation of OFMTutor, an Operator Function Model intelligent tutoring system. OFMTutor is intended to provide intelligent tutoring in the context of complex dynamic systems for which an operator function model (OFM) (Mitchell, 1987) can be constructed. The human operator's role in such complex, dynamic, and highly automated systems is that of a supervisory controller whose primary responsibilities are routine monitoring and fine-tuning of system parameters and occasional compensation for system abnormalities (Sheridan and Johannsen, 1976; Wickens, 1984).

The ability of a supervisory controller to cope with abnormal or emergency situations can be severely limited. Wickens (1984) cites several problems with supervisory control: an increased monitoring load; a "false sense of security" whereby the operator trusts the automation to such an extent that any human intervention or checking seems unnecessary; and "out-of-the-loop familiarity" that implies a reduced ability to cope with non-routine situations.

An important question then becomes how to improve system performance and safety in supervisory control. The answer is not to automate the human out of the system; today's technology cannot match the human's ability to cope with uncertain and novel situations (Chambers and Nagel, 1985). Rather, automated systems must support the human operator.

One potentially useful form of support is the use of intelligent tutoring systems to teach the operator about the system and how to function within that system. In the next section, previous research on intelligent tutoring systems (ITS) is considered. Then the proposed design for OFMTutor is presented, and an experimental evaluation is described.

INTELLIGENT TUTORING SYSTEMS RESEARCH

Intelligent tutoring systems are usually described in terms of three modules: an expert module that represents domain expertise; a student model that represents the student's performance record and presumed state of knowledge; and a tutorial module that structures the interaction between the tutor and the

student (Fath et al., 1988; Park et al., 1987; Wenger, 1987). Wenger (1987) considers the interface to be a fourth critical component for the successful implementation of a knowledge communication system. The following discussion of ITS research is divided into these broad categories of domain expertise representations, student modeling, pedagogical strategies, and interface design. Particular attention is paid to efforts that involve complex dynamic systems and/or complex problem solving tasks.

Domain Expertise

According to Wenger (1987), domain expertise forms the source of knowledge to be communicated and the standard for evaluating performance. Thus, the teaching goal is explicitly represented as this knowledge. The degree to which such domain expertise may be articulated is dependent upon the transparency of the expert model's structure and its psychological plausibility. Thus, knowledge whose structure is transparent to the student and whose organization and form are psychologically plausible is knowledge that can be relatively easily communicated to the student.

The SOPHIE project involved a series of tutoring programs for electronics troubleshooting (Burton et al., 1982; Wenger, 1987). The first version of SOPHIE (SOPHIE-I) demonstrates the efficiency and robustness of an inference engine that uses multiple representations of domain knowledge: a simulation-based mathematical model of a circuit; procedural knowledge, organized as a collection of specialists, to act on the circuit model; and declarative knowledge organized as a semantic net of facts. SOPHIE-I was used as a supplemental laboratory in electronics troubleshooting instruction. SOPHIE-II extends SOPHIE-I to include an articulate troubleshooting expert to demonstrate strategies to the student. The emphasis is on articulation of expertise in qualitative, causal terms. Finally, SOPHIE-III is meant to support learner-centered activities, while providing powerful inference capabilities and supporting good explanations. SOPHIE-III's expertise is represented as two separate modules: a troubleshooting expert and an electronics expert. The architecture supports flexible, humanlike reasoning.

Like SOPHIE, the Recovery Boiler Tutor (RBT) (Woolf, 1986) supports a "reactive learning environment" which includes a simulation of the system of interest (a kraft recovery boiler) and in which the student is allowed to propose hypotheses that can be evaluated in real time. Also like SOPHIE, RBT's domain

expertise concerns fault detection and diagnosis. The domain knowledge is represented as a knowledge base of scenarios which describe preconditions, postconditions, and solutions for emergencies or operating conditions.

The AHAB system represents the realization of a proposed ITS architecture for troubleshooting in complex dynamic systems (Fath, 1987; Fath et al., 1988). AHAB works in conjunction with PEQUOD, a marine steam powerplant simulation, to tutor students in troubleshooting strategies. AHAB's domain expertise (a "task model") prescribes troubleshooting actions based on current system state and represents psychologically plausible troubleshooting strategies (i.e., symptomatic and topographic search (Rasmussen, 1986)). The task model is structured as an operator function model (OFM) (Mitchell, 1987), together with representations of declarative and procedural knowledge.

AHAB's OFM provides the richest, most efficient structure for domain expertise of all three systems reviewed here. It accounts for the coordination of strategy and dynamic focus of attention based on current system state. The OFM will be described in more detail in the section on the design of OFMTutor.

Wenger discusses several issues in the representation of domain expertise. The representation should be complete; that is, the expertise should be a process model that has knowledge of the domain and also metaknowledge about how to use it. A process model must be able to solve problems that the student is expected to learn.

Domain knowledge also needs to be relevant to the student. To this end, the process of warranting belief (i.e., justification of new knowledge with respect to previous knowledge and beliefs) is important. It is crucial to motivate the concept to be taught with references to previous knowledge and beliefs held by the student. This serves to justify new knowledge.

Finally, a critical distinction in domain knowledge is whether it is compiled or articulate. Compiled knowledge is "automatic"; it is efficient and simple to use, but no longer possesses transparency and generality. In particular, compiled knowledge does not support the warranting process. Articulate knowledge, on the other hand, is able to support warranting belief via *organization* in terms of decomposition into primitives and configuration of primitives into a model and *justification* in terms of "first principles" (e.g.,

causality, structure, functionality, teleology) and integration.

Student Modeling

Intelligent communication requires understanding of the recipient (Wenger, 1987). Thus, an intelligent tutoring system must possess some model of the student's current state of knowledge. In particular, an ITS uses student actions as data for interpretation and reconstruction of presumed states of knowledge. This requires the explicit consideration of a model of the student. The student model is needed by the tutor to guide the student's problem solving and to organize the learning sequence (Wenger, 1987).

In general, student modeling employs the technique of "differential modeling" -- that is, a comparison of expert and student performance and/or knowledge (Burton and Brown, 1982). Differential modeling requires that the expert and student models have the same structure for unambiguous comparisons to occur. Student modeling techniques may be broadly classified as overlay models or buggy models (Park et al., 1987; Wenger, 1987). An overlay model assumes that the student's knowledge is a subset of expert knowledge; differences between the student and expert models are due to incompleteness of student knowledge. An example of an overlay model is Goldstein's genetic graph (Goldstein, 1982). Buggy models explicitly capture misconceptions as a collection of "bugs" (with or without an accompanying theory of the origin of these bugs); differences between the student and expert models are due to the student's "buggy" deviations (Burton, 1982; Johnson and Soloway, 1985). Buggy models are employed in BUGGY and its variants (Burton, 1982) and PROUST (Johnson and Soloway, 1985).

Two particular student modeling techniques are relevant for our discussion. The first is the idea of the "limited bug model" used in AHAB. AHAB's student model is very similar in structure to the task model, and thus the two can be compared via differential modeling in the spirit of an overlay model. However, AHAB also represents student errors in terms of common or important general types of errors. Thus, errors are not exhaustively enumerated a priori, but broad categories of errors can be used to identify the source of a difference between the student and task models.

The second important consideration is that of student intentions. By utilizing an explicit account of plausible student intentions (i.e., goals and plans), performance can be better understood and thus

diagnosed properly and remedied in context (Genesereth, 1982).

> Plan recognition is a way of using information about the student's actions in dealing with the combinatorics in domains where the number of reasonable solutions and bugs is too large for the expert difference technique to work effectively. ... In addition to helping pinpoint the student's misconception, studying his plan is advantageous in that it enables the tutor to offer remediation in the context of the student's problem and his approach to solving it. (p. 140)

Similarly, Johnson and Soloway (1987) argue that "knowledge of intentions makes it possible to identify more bugs, as well as to understand their causes" (Johnson and Soloway, 1987, p. 50). Thus, it is desirable to account for plausible student intentions explicitly in the design of a student model.

Pedagogical Strategies

Pedagogical strategy defines the organization, sequencing, and form of the student-tutor interaction; it designates what to say and how and when to say it. Many intelligent tutoring systems employ the guided discovery or coaching method in which the student "learns by doing" with the assistance of a non-intrusive coach (Park et al., 1987; Burton and Brown, 1982; Wenger, 1987). The purpose of the coach is to "foster the learning inherent in the activity itself by pointing out existing learning opportunities and by transforming failures into learning experiences" (Wenger, 1987, p. 124).

WEST is one of the earliest and most influential computer coaches. WEST assists students in playing the game "How the WEST was Won" (Burton and Brown, 1982). WEST uses the "issues and examples" paradigm to find issues where a student is weak (via differential modeling) and then to provide examples to illustrate better moves. The guiding principle behind such a pedagogical strategy is to make interventions both relevant and memorable (Wenger, 1987). WEST also employs a number of tutoring principles that govern its intervention capabilities (e.g., "Never tutor on two consecutive moves") (Burton and Brown, 1982).

SOPHIE and RBT both provide a "reactive learning environment" in which students can "play" with the simulation and observe the effects of their manipulations. STEAMER is an inspectable, interactive simulation of a propulsion plant that also allows students to manipulate system conditions and events (Hollan et al., 1987). Fath (1987; Fath et al., 1988) explicitly considers the simulation of a complex system as part of the instructional media; the simulation is important in supporting students' understanding of the system (i.e., building of accurate mental models (Hollan et al., 1987; Wenger, 1987)). Such simulations are important pedagogically in that they can serve as a form of continuous explanation to the student (Wenger, 1987).

An important facet of pedagogy is diagnosis of student misconceptions. Diagnosis updates the student model to reflect issues that need to be addressed in the interaction. Wenger (1987) distinguishes between three levels of diagnostic activities: behavioral, epistemic, and individual. Behavioral diagnosis is concerned with behavior and the product of behavior. It is further characterized as *non-inferential classification* (an evaluation of student performance in terms of correctness) or *inferential reconstruction* that is concerned with the problem solving process. The latter form of behavioral diagnosis is of concern here; inferential reconstruction deals with the use of plans and goals in reconstruction of problem solving behavior. Of especial importance are the PROUST and MACSYMA Advisor systems that explicitly represent student intentions. High-level goals are decomposed into plans and actions; diagnosis is a process that alternates "between model-driven confirmation and data-driven recognition of plans and goals" (Wenger, 1987, p. 374).

An important issue in instructional systems in general is principled curriculum design. While the so-called "frame-based" computer-based instructional methods made an effort to take these considerations into account, much of the research in intelligent tutoring systems does not make use of a theory of learning or instruction (Lesgold, 1988; Park et al., 1987). However, some efforts have been made to consider curricula in ITS research. Wenger (1987) discusses the concept of a *bite-sized tutoring architecture* (proposed by Bonar and his colleagues) in which the system is organized around pedagogical issues called bites. Each bite focuses on a particular aspect of domain knowledge and also includes knowledge of its conceptual and

curricular relations to other bites, the student's mastery of that bite's subject matter, and the abilities to diagnose, generate problems, and generate instructional interventions.

Lesgold (1988) points out that

> Where conventional instruction has an explicit curriculum but fails to have an explicit and complete representation of the knowledge that is to be taught, intelligent instructional systems have tended to represent the target knowledge explicitly but not to represent explicitly that body of knowledge that specifies the goal structure for instruction, the curriculum. (p. 117)

Lesgold argues that an intelligent tutor must represent domain knowledge, curriculum knowledge, and knowledge of metaissues that affect instruction. Domain knowledge includes procedural and declarative (i.e., conceptual) knowledge. Curriculum knowledge is represented as a lattice of goals that are decomposed progressively into subgoals, down to lessons that can be taught completely as a unit. This structure is based upon Gagne's (1971) learning hierarchy, in which the goal of instruction is progressively refined down to the level of individual lessons. The curriculum goal lattice is composed of a number of such goal hierarchies, each of which corresponds to a particular viewpoint of domain knowledge. Finally, the metaissue layer relates to knowledge of student aptitude (e.g., "good at math") and is defined as the topmost goal nodes in the curriculum lattice (i.e., the origins of the various viewpoint hierarchies).

Lesgold emphasizes that the implicit idea of Gagne's learning hierarchy is that the whole is more than the sum of its parts; higher levels in the hierarchy also provide "conceptual glue" that relates lower level knowledge. Furthermore, he argues that the knowledge taught in a lesson depends upon the context in which the lesson is taught. When a lesson is first taught, its "core content" (i.e., a coherent subset of knowledge) should be presented, but if a lesson is remedial, "it is crucial to teach the knowledge that links the core content of the to-be-remediated lesson with the core content of the lesson whose failure produced the need for remediation" (Lesgold, 1988, p. 134).

Gagne's ideas have had a large influence on instructional design practices. He and his colleagues have developed a well-specified approach to instructional design which can be summarized as follows (see Briggs, 1977a):

1. Needs Assessment

This is a process of identifying instructional goals, ranking them by importance, identifying one or more needs, and setting priorities for action.

2. Write Performance Objectives

Performance objectives translate goals into specific behavioral criteria for successful performance. The proposed Gagne-Briggs model of performance objectives distinguishes between action, object, situation, tools and other constraints, and the capability to be learned. The action denotes what observable behavior the student will perform (e.g, writing, running). The object denotes the resulting product of the action (e.g., a poem, a painting). The situation describes the circumstances in which the student will perform (e.g., given the PEQUOD simulation with one introduced fault). Tools and other constraints describe how the action will be carried out (e.g., with a pencil) and performance limits (e.g., without the use of references, within 30 minutes). The capability to be learned is inferred from the action; Gagne and Briggs have proposed a taxonomy of capabilities that distinguishes between intellectual skill, cognitive strategy, information, motor skill, and attitude. This taxonomy is shown in Table 1.

Insert Table 1

about here

An example of a problem-solving performance objective is as follows (see Kibler and Bassett, 1977). Suppose the domain of interest is instructional design, and students are to be taught how to write performance objectives. A reasonable performance objective might be: Given a general statement of the scope and

sequence of topics for a high school course (situation), generate (learned capability: problem solving) appropriate student objectives (object) by writing such objectives (action) within one week (tools, constraints, and special requirements).

3. Analyze Objectives

This is a kind of task analysis in which the taxonomy shown in Table 1 is used. The learning task is analyzed with respect to its essential and supporting prerequisites; such prerequisites define a learning hierarchy. For example, an intellectual skill has as essential prerequisites simpler component intellectual skills and as supporting prerequisites (i.e., those not essential for learning but that can be helpful) attitudes, cognitive strategies, and verbal information.

4. Design the Instructional Strategy

The learning hierarchy provides a prescription of how the instruction should be sequenced; prerequisite skills should be taught first. In other words, instruction proceeds "bottom up". At a more fine-grained level, Gagne distinguishes between nine instructional events (or teaching steps): gain attention, tell the student the objective, stimulate recall of prerequisites, present the stimulus material, provide guidance, elicit the performance, provide feedback, assess performance, and enhance retention and transfer. Of particular interest are Gagne's suggested forms of guidance for learning; these are reproduced in Table 2.

Insert Table 2

about here

---

5. Lesson Planning

The suggested steps in lesson planning are to identify the objective, list the desired instructional events, select ideal media, select materials and activities, analyze materials for events they supply, and plan other means for the remaining events (Briggs, 1977b).

6. Formative Evaluation

Formative evaluation is the process of testing and revising instructional materials while they are still being developed. Three suggested phases of formative evaluation are one-to-one, small group, and field trial evaluations (Dick, 1977a).

6. Summative Evaluation

Summative evaluation is the process of collecting and interpreting data about the quality of a proposed educational product. A suggested five-step approach to summative evaluation is to identify instructional objectives, identify the target population and select students from it, develop evaluation instruments (i.e, objectives-referenced tests, attitude questionnaires, and cost data), document the instructional process, and prepare the final report (Dick, 1977b).

This approach has been quite popular in the educational community; however, it is not without its critics. Novak (1986) argues that Gagne's model of learning (i.e., the learning hierarchy) is founded on a stimulus-response association. As such, this model, with its emphasis on "behavioral objectives" and suggested bottom-up approach to instructional sequences, is an outgrowth of behaviorist psychology. Novak argues for a constructivist, rather than a positivistic, view of epistemology. In other words, rather than an exclusive concern with observable data, we should recognize that "humans construct knowledge using the concepts, principles, and theories they have, and change their *knowledge claims* as new ideas and associated methodologies lead to new constructions of how people and the universe operate" (Novak, 1986, p. 6). Novak also emphasizes the importance of *concepts;* in fact, he states that "'concepts are what we think with.' As we change our concepts and conceptual frameworks in positive ways, we may or may not change our *behavior*, but the *meaning* of our experience changes" (Novak, 1986, p. 8).

Novak argues that a model of human learning is essential for a theory of education. Rather than a behaviorist model of learning, he advocates the theory proposed by educational psychologist David Ausubel. Ausubel contends that the single most important factor in learning is what the learner already knows. Effective instruction ascertains the learner's present state of knowledge and teaches accordingly. Ausubel

views knowledge as a *cognitive structure:* a hierarchical organization of concepts, where specific elements of knowledge are subsumed under more general concepts.

Another important idea is that of concept differentiation: "As new experience is acquired and new knowledge is related to concepts already in a person's mind, these concepts become elaborated or altered, and hence they can be related to a wider array of new information in subsequent learning" (Novak, 1986, p. 25). Thus, a learner's previous knowledge includes relevant concepts and the extent of their differentiation.

Ausubel distinguishes between rote and meaningful learning. Rote learning occurs when the new material is not associated with any existing elements in the cognitive structure. Meaningful learning occurs when new material is linked with subsuming concepts ("subsumers"). The new material is "stored in a somewhat altered form (as a product of assimilation with the subsuming concept(s)) and modifies (differentiates further) the subsumers to which it is linked" (Novak, 1986, p. 26).

Ausubel's theory implies that "concept development proceeds best when the most general, most inclusive elements of a concept are introduced first and then the concept is progressively differentiated in terms of detail and specificity" (Novak, 1986, p. 86). This is in direct contradiction to Gagne's prescription of teaching "bottom up". Ausubel's theory is more persuasive, however, in that "top down" instruction gives a context for learning, and, in Wenger's terms, may serve to warrant belief.

Novak emphasizes the distinction between curriculum (issues) and instructional (teaching) issues. This is similar to the separation of domain and tutorial knowledge that distinguishes the GUIDON system (Clancey, 1987). Novak draws on Johnson's model of curriculum design (Johnson, 1967), a simplified version of which is shown in Figure 1.

Insert Figure 1

about here

It is shown that a curriculum development system uses knowledge available from the culture in conjunction

with structuring and selection criteria to produce a curriculum. The curriculum is an input to structured planning, which also considers instrumental content and teaching behavior repertoire to produce an instructional plan. This plan is administered to learners; their performance is evaluated and relevant feedback is provided to both curriculum and instructional development.

Novak relates Ausubel's theory of learning to Johnson's model of curriculum and instructional development in order to specify the nature of each module. Ausubel's emphasis on concepts implies that the selection criteria for knowledge should be to select major and minor concepts in the field of study. The ordering criteria for knowledge should consider both progressive differentiation and integrative reconciliation:

> Meaningful learning and progressive differentiation require the most general, most inclusive concepts be presented early and subsequent information be provided to clarify meaning and show connections to subordinate concepts....Superordinate learning and integrative reconciliation require that subordinate concepts be presented in a manner that allows association with more inclusive concepts (superordinate concepts), and meanings of apparently disparate concepts will be clarified to show distinctions and relationships between subordinate concepts (integrative reconciliation). (Novak, 1986, pp. 137-138)

Ausubel's theory implies that the curriculum's "intended learning outcomes" (ILOs) should be the concepts to be learned, with associated hierarchical and subordinate relationships. The selected exemplars should be chosen such that "cognitive bridging" is provided (i.e., explicit association between new concepts and the existing cognitive structure). Teaching approaches should to flexible and allow for "hands-on experience" (or, in Piaget's terms, experience with "concrete props"). Actual learning outcomes are a function of the "degree of overall cognitive structure differentiation" and "initial or developed relevant subsumers in the learner's cognitive structure" (Novak, 1986, p. 139). Evaluation can be examined in terms of rate or degree of transfer of learning. The rate of learning depends on the "quality of existing or developed relevant subsumers, and motivation for learning. Transfer of learning to new problem solving situations

will be a function of the degree of concept differentiation, superordinate subsumption, and integrative reconciliation achieved." (Novak, 1986, p. 139). The feedback to curriculum planning may imply the need for "alternative sequences of concept presentation" or "better clarification of relationships between concepts...and/or better description of salient aspects of the concept(s)" (Novak, 1986, p. 139). Finally, the feedback to instruction may indicate the need to select better exemplars (i.e., select those more easily linked to existing cognitive structure), provide better pacing of instruction, or select a better instructional strategy (e.g., one-on-one tutorial assistance when the learners' cognitive development is highly variable).

As will be described later, it is proposed that OFMTutor employ the coaching method of tutoring, with intention-based diagnosis, a modified version of Lesgold's architecture, and an emphasis on relevant, meaningful concepts as argued by Ausubel and Novak.

Interface Design

The interface design is an important part of an intelligent tutoring system, for the student directly experiences interaction with the interface. Two basic routes have been taken in the design of the interface: one concerned with graphical, iconic representations; the other with (textual) dialogue management. Often, graphical representations are used in conjunction with guided-discovery learning or coached activities (e.g., STEAMER, IMTS, RBT), and dialogue systems employ a mixed-initiative dialogue style of interaction (e.g., GUIDON).

Graphical Interfaces. In the domain of complex, dynamic systems, dynamic graphical representations of the system's structure and function are useful. STEAMER's developers felt that a graphical interface to a simulation would be valuable in that it would allow one to view and manipulate the system at a number of different hierarchical levels. The graphical interface is also meant to provide a conceptually faithful representation of the system, in order to foster the development of accurate mental models of the system (Hollan et al., 1987).

The IMTS system provides a number of software tools that enable the development of simulation-based technical training (Towne et al., 1988). IMTS supports both physical and functional views of device components, in order to "promote the student's ability to find, recognize, and manipulate physical elements

in the real system, while maintaining a conception of the functional relationships that cannot be seen directly in the real system" (Towne et al., 1988, p. 18).

RBT "provides tools for reasoning" about the operation of a kraft recovery boiler. These tools include graphs that depict the relationship between various process parameters over time, meters that show the system state at higher levels of abstraction (e.g., safety, efficiency, and reliability), and interactive tutorial dialogues. The system also uses animated graphics to represent a mathematically and physically accurate simulation of the boiler.

AHAB works with the PEQUOD simulation of a marine steam powerplant. PEQUOD uses a qualitative approximation methodology (Govindaraj, 1987) that represents the system at various hierarchical levels. System states are calculated quantitatively but gauge readings, etc. are represented qualitatively. The student can inspect schematics of subsystems that provide information on causal flow and system state. The student also interacts with menus and windows to determine feasible tests, make diagnoses, and gain performance feedback.

GUIDON-WATCH is a graphic interface to NEOMYCIN, a medical consultation system. GUIDON-WATCH allows the student to browse through the database and view reasoning processes for diagnosis (Richer and Clancey, 1985). A number of windows provide information on current hypotheses, causal relations among symptoms, a diagnostic task tree, current evidence, and positive findings.

The importance of a graphical interface is supported by research in human-machine interaction (cf Norman and Draper, 1986). Some forms of interaction are facilitated with a conceptually natural and simple interface composed of iconic representations of objects rather than text. For training and tutoring in domains associated with complex dynamic systems, graphical interfaces allow the student to conceptualize system structure and function in a natural manner and to concentrate on learning rather than the interaction itself.

Toward a Theory of Discourse. When tutorial interactions are necessary, how should the tutor intervene? How does the tutor know when to intervene, what to say, and how to say it? The answer to these questions involves a consideration of discourse, which is intimately tied to the chosen pedagogical strategy.

Burton and Brown (1982) provide specific principles of interaction for a tutorial coach; these are summarized in Table 3.

Insert Table 3

about here

Such rules of thumb are useful guidelines but do not provide any principled theory of discourse conventions in tutoring. Other researchers have focused on how human tutors interact with students (Woolf, 1987; Fox, 1987a and 1987b). Fox (1987a) argues that the tutorial interaction itself (e.g., the way the student responds to questions and the timing of the response) provides diagnostic information and advocates that at least timing information can be utilized by an automated tutor. Thus, a lengthy pause between a question posed by the tutor and the student's response may indicate that the student encounted difficulty in arriving at the answer. Fox also stresses that "repair is an essential factor in natural language interface design" (Fox, 1987b, p. i). In an analysis of face-to-face conversational turn-taking between a human tutor and student, Fox notes that such turn-taking is very flexible, not primarily controlled by the tutor, and offers a fundamental mechanism for repair that is missing from the same type of interaction over a teletype machine. What is lost from the terminal-to-terminal interface is the "opportunity for the hearer to indicate understanding or lack of understanding at the end of every unit; and the opportunity for the speaker to initiate correction on his/her own turn after it was sent" (Fox, 1987b, p. 5). Fox further notes that "certain kinds of interruption are essential for maintaining mutual comprehension" and thus vital for repair (Fox, 1987b, p. 8). Fox's suggestions for dialogue management for an intelligent tutoring system are given in Table 4.

Insert Table 4

about here

Clancey (1982) argues that a case method tutor needs knowledge of dialogue patterns, domain knowledge, and the communication situation in order to carry on a dialogue with a student. Clancey's case method tutor is GUIDON, a tutor that works in conjunction with MYCIN, an expert system for diagnosis of infectious diseases. Augmented domain knowledge allows GUIDON to use metaknowledge to reason about MYCIN's rules and thus to use domain rules in a variety of ways. The communication situation is defined by the student model (an overlay model that represents what topics or rules the student has and has not yet demonstrated that he/she has learned) and the "focus record" that lists goals that the student has inquired about. GUIDON uses "discourse procedures" invoked by tutoring rules to direct and focus the case dialogue. The tutoring rules use knowledge of the communication situation as preconditions; thus, the communication situation drives the tutorial interaction. The tutoring rules are used to select discourse patterns (guide discussion of a domain rule, respond to a student hypothesis, and choose question formats), choose domain knowledge (provide orientation for choosing new goals, measure interestingness of domain rules), and maintain the communication model (update the student model).

Woolf (1987) has investigated the machine representation of discourse conventions in tutoring. She notes that discourse "is often described in qualitative terms along with the *effect* of the utterance on the listener" (Woolf, 1987, p. 250). Discourse analysis often seems guided by implicit rules that are based on the perception of qualitative states such as "what the student already knows". Woolf and her colleagues have begun developing a theory of discourse based on the recognition of qualitative states such as "student is confused" and "topic is generally known". As a first step towards this, Woolf defines conversational *move-classes* "as groups of utterances that have the same rhetorical effect" (Woolf, 1987, p. 253), such as question-topic and provide-example. The choice of a move-class indicates the speaker's intention in that the listener has particular expectations given a certain type of move. Woolf proposes tutoring maxims based on Paul Grice's maxims of conversation: quality (be truthful), quantity (be brief, yet complete), relation (be relevant) and manner (be clear and orderly) (Mura, 1983). Woolf's adaptions of Grice's maxims are shown in Table 5, and the tutorial maxims in relation to move-classes are shown in Table 6.

Insert Tables 5 and 6

about here

The move-classes are also defined with respect to their probable implications. These implications define implicit assumptions that are "taken for granted" in natural conversation. For instance, the choice of the question-topic class implies that the tutor "knows (or attempts to learn) the student's threshold of knowledge", "assumes the student can answer the question", and "thinks the topic is important or is learnable through the discourse" (Woolf, 1987, p. 256). Additional global implications are possible, based on "extended reasoning about sequences of move-classes" (Woolf, 1987, p. 256). Such global implications are assessments such as "student understands" and "topic was complete"; these are inherently uncertain and form the system's current best hypothesis of the student's state of knowledge or current topic. Such reasoning with uncertainty requires the ability to entertain multiple, possibly conflicting hypotheses and to resolve conflicts based on accumulating evidence.

Summary

This section has discussed previous research in intelligent tutoring systems, with emphasis on applications to domains associated with complex dynamic systems. Theories of education, learning, and discourse have also been considered. The following considerations are especially relevant to the design of OFMTutor: the simulation of a complex dynamic domain, a process model of expertise, intention-based diagnosis and student modeling, the coaching/guided discovery paradigm, the importance of concepts in a structured curriculum, a graphical dynamic interface to a complex dynamic simulation, and discourse models that allow for repair.

OFMTUTOR DESIGN

The design philosophy behind OFMTutor is similar to that of the RBT system (Woolf, 1986): the tutor is a "partner and co-solver of problems with the operator" (Woolf, 1986, p. 11). This has much in common with the approach of the "joint cognitive system" described by Woods (1986). The joint cognitive

system paradigm proposes that the computer provide support for the operator, giving context-sensitive reminders and suggestions, rather than dominate the interaction.

This approach has several implications for the design of an intelligent tutoring system for supervisory control of a complex dynamic system. First, the domain expertise and student models must be represented as *process models* (Wenger, 1987) that explicitly capture procedural knowledge and decision making behavior. Second, joint hypothesis formation and decision making imply the need for a pedagogical strategy of coaching in a guided discovery (reactive learning) environment. Finally, the interface design must support the explicit representation of domain expertise and an inspectable model of joint hypotheses.

In the next section I review the theoretical foundations of OFMTutor with a discussion of the operator function modeling methodology. Next, a blackboard architecture that implements the OFM for intent inferencing is discussed. Finally, the design of OFMTutor is presented. The OFMTutor architecture is based on that of OFMspert (Operator Function Model expert system) (Rubin et al., 1987).

The Operator Function Model

The OFM provides a flexible framework for representing operator functions in the control of a complex dynamic system. The OFM represents how an operator might organize and coordinate system control functions (Mitchell, 1987). Mathematically, the OFM is a hierarchic-heterarchic network of finite-state automata. Network nodes represent operator activities as operator functions, subfunctions, tasks, and actions. Operator functions are organized hierarchically as subfunctions, tasks, and actions. Each level in the network may be a heterarchy, i.e., a collection of activities that may be performed concurrently. Network arcs represent system triggering events or the results of operator actions that initiate or terminate operator activities. In this way, the OFM accounts for coordination of multiple activities and dynamic focus of attention. A generic example of an OFM is illustrated in Figure 2.

Insert Figure 2

about here

Historically, the OFM is related to the discrete control modeling methodology (Miller, 1985; Mitchell and Miller, 1986). The OFM is distinguished by its modeling of both manual and cognitive operator actions in the context of system triggering events. Manual actions are system reconfiguration commands. Cognitive actions include information gathering and decision making that are typically supported by information requests.

The OFM is a prescriptive model of human performance in supervisory control. Given system triggering events, it defines the functions, subfunctions, tasks, and actions on which the operator should focus. Used as an expert model, the OFM generates expectations of likely operator actions in the context of current system state. Used as a student model, the OFM defines likely operator functions, subfunctions, and tasks that can be inferred based on operator actions and system state. This ability to infer intentions dynamically is crucial for intention-based diagnosis, and also forms the core of the domain expertise and student models.

Successful application of the OFM to intent inferencing in a supervisory control task (Rubin et al., 1987; Jones, 1988; Jones et al., 1988) demonstrates that the OFM is a viable basis for determining operator (student) intentions in the context of current system state and past operator actions. This application utilized a knowledge-based problem solving methodology known as the blackboard model of problem solving (Nii, 1986). The next section describes this implementation.

The Blackboard Model of Problem Solving

The blackboard model of problem solving consists of three components: the blackboard, knowledge sources, and blackboard control (Nii, 1986). The blackboard is a data structure on which the current best hypothesis of the solution is maintained and modified. The hypothesis is represented hierarchically, at various levels of abstraction, and evolves incrementally over time as new data become available or old data become obsolete. Domain-specific knowledge is organized as a collection of independent knowledge sources. Knowledge sources are responsible for posting and interpreting information on the blackboard.

Blackboard control applies knowledge sources opportunistically; that is, in either a top-down or bottom-up manner, depending on what is more appropriate in the current context.

The blackboard model of problem solving is compatible with the knowledge represented in the OFM. Both models use a hierarchical representation. The blackboard knowledge sources provide a modularity that naturally represents much of the domain knowledge contained in the OFM arcs. The opportunistic control strategy offers the dynamic flexibility necessary for inferring intentions in real time.

Operator intentions may be represented as a hierarchy of goals, plans, tasks, and actions that correspond to the OFM's hierarchy of functions, subfunctions, tasks, and actions. Goals are currently instantiated functions, plans are currently instantiated subfunctions, and so on. The general mechanism for the blackboard approach to intent inferencing is as follows. Given an OFM, currently hypothesized goals, plans, and tasks (GPTs) or sometimes additional plans and tasks (PTs) for an existing goal are placed on the blackboard in response to system triggering events. The blackboard incorporates operator actions into the representation with opportunistic reasoning. Thus, actions can be immediately interpreted as supporting one or more current goals, plans, and tasks; and goals, plans, and tasks can be inferred on the basis of operator actions. In general, actions are interpreted with a strategy of maximal connectivity; actions that can support more than one current task are interpreted as supporting all such tasks. Figure 3 shows the proposed blackboard model of intentions.

Insert Figure 3

about here

Other Modeling and Pedagogical Considerations

The OFM is not enough to define all the knowledge needed for tutoring. The OFM does not explicitly represent the system to be controlled; it specifies operator functions within that system. This level of explanation is reasonable for well-trained operators but is insufficient for tutoring purposes. Like

GUIDON, OFMTutor will have to be augmented with supporting domain knowledge. Based on Novak's theory of education, such supporting knowledge should be in the form of concepts whose subsuming relationships should be clearly explicated. The relevant concepts associated with a complex dynamic system include the system's purpose, function, and structure. In particular, these concepts can be described in accordance with Rasmussen's (1986) abstraction hierarchy. Thus, both knowledge of the system (the abstraction hierarchy) and knowledge of operator functions (the OFM) correspond to a hierarchical arrangement that suggests the course of "top down" instruction proposed by Novak. Furthermore, since knowledge of the system is, in Gagne's terms, an essential prerequisite for knowledge of how to control the system, system concepts should be taught before operator function concepts. Figure 4 illustrates the abstraction hierarchy.

Insert Figure 4

about here

Furthermore, the OFM does not represent errorful behavior. Thus, the inclusion of "buggy" GPT's is necessary as a "limited bug model" similar to AHAB's. In this way, broad classes of misconceptions can be diagnosed and remedied appropriately. In order to build representations of misconceptions, a thorough cognitive task analysis of novice users is necessary. By careful observation of subjects interacting with the system, as well as protocol and off-line analyses, one may be able to characterize misconceptions and their manifestations in action patterns. Then the intelligent tutoring system can be given knowledge of particular action sequences and probable underlying misconceptions associated with them.

The Representation of the Expert and Student

The combination of the OFM and blackboard model of problem solving define a process model (Wenger, 1987) that can be used to represent expertise and student knowledge. The "expert's" goals, plans, and tasks can be represented on one blackboard, and the student's inferred intentions can be represented

similarly on a separate blackboard. Differential modeling can easily assess the difference between the two blackboards. If the student model is missing intentions that the expert has, a reminder or hint can be employed to remedy this error of omission. If the student model includes some intentions not modeled on the expert blackboard, this may signal a misconception to be corrected with the proper intervention.

The expert's goals, plans, and tasks are inferred on the basis of current system state. Thus, intentions are derived from the (normative) operator function model for GT-MSOCC. In contrast to the expert's model-derived intentions, the student's goals, plans, and tasks are inferred based on student actions. The student's intentions may also be modeled with the "buggy" GPT's described previously. Both representations exist in a component of OFMTutor known in general as the Blackboard. The Blackboard actually consists of an expert and a student blackboard, where hypothesized intentions are posted and compared.

Supporting Domain Knowledge

Like AHAB and RBT, OFMTutor will require that students interact with a simulation of a complex dynamic system. Unlike AHAB and RBT, OFMTutor is designed to exist on a computer separate from the simulation itself. This distributed environment supports the clean separation of domain dynamics and tutoring knowledge and strategy. Furthermore, the portability of the OFMTutor architecture is enhanced in that it can, in theory, be placed "on top of" any complex dynamic system simulation for which an OFM can be constructed. Philosophically, it can be argued that intelligent tutoring is but one point on a continuum of intelligent aiding in general, and since our design philosophy of the operator's associate dictates such a distributed environment (Rubin et al., 1987; Jones et al., 1988), it is natural that OFMTutor also requires such an architecture.

The requirement of a distributed environment means that OFMTutor must have an explicit representation of current system state. This representation may be termed the Current Problem Space (CPS). The CPS is needed to give context for the modeling of intentions and for pedagogical interventions. Also, a representation of the current displays to the student is necessary in order to infer what information is currently available to the student. This representation is called the Workstation description. The Workstation maps the names of display pages to their semantic information content.

OFMTutor must also support knowledge of domain concepts. Beyond the procedural knowledge defined in the OFM, an operator must have some grasp of the underlying principles of the system's structure and function. This includes knowledge of the system's purpose, abstract function (e.g., flow of data), function, and physical form. Such knowledge can be captured within the framework of Rasmussen's (1986) abstraction hierarchy. The abstraction hierarchy forms part of the additional knowledge needed for tutoring. This additional knowledge also includes pedagogical knowledge and strategies and the limited bug model goals, plans, and tasks. These enhancements to the knowledge in the OFM define the Enhanced Normative Model (ENM).

Finally, the architecture requires a Communication Interface that communicates with the simulation of the system and with the tutorial interface to the student. Information about system events and student actions is sent from the simulation and the tutorial interface to OFMTutor. A scheduler, called the High Level Controller, manages the various events within OFMTutor. The complete architecture is shown in Figure 5.

Insert Figure 5

about here

Pedagogical Strategy

OFMTutor's pedagogical strategy is guided both by Ausubel's and Gagne's ideas; i.e., concepts and essential prerequisites. Since a concept of the system is an essential prerequisite for learning operator functions to control that system, the instructional process is divided into two broad sections: one on concepts, and one on control. The "conceptual" curriculum is presented top down and organized with respect to Rasmussen's abstraction hierarchy. The student's knowledge is assessed via on-line quizzes and questionnaires.

When the student has mastered system concepts, control functions will be taught. Here, the student will learn procedures for controlling the system. This phase of instruction is organized top down with respect to the operator function model. Also, while the first phase of instruction will take place solely in the context of the tutor, the second "operational" phase is taught in the context of the system simulation. In this phase, the tutorial strategy is one of guided discovery, in which the student explores the system and is given non-intrusive assistance by a "coach." The student has relatively greater control over the interaction here than in the first phase.

Diagnosis occurs by differential modeling of the two blackboards as described earlier. However, the tutor does not give explicit advice or warnings immediately when a discrepancy is noticed. It is important to allow an opportunity for the student to "get back on track" independently.

Tutorial Interface

OFMTutor's interface is a multiwindow environment that allows the student a fair degree of control over the interaction. The student may collapse or open any windows desired at any point in time. In the first phase of instruction, the tutorial interface consists of animated views of the system, text that describes concepts, and multiple-choice and fill-in-the-blank quizzes.

During the second phase of instruction, the interface supports a graphical representation of joint intentions; that is, a representation that explicitly shows the comparison of the expert and student model blackboards. Supporting text windows include a list of expected commands, a list of all commands (so that failures to remember syntax are minimized), and dynamically generated advice and suggestions.

OFMTUTOR EVALUATION

Park et al. (1987) assert that one methodological difference between computer-based instruction and intelligent tutoring systems is that the former pay attention to evaluation procedures, and the latter do not. In this section we examine evaluation procedures from both instructional and industrial-organizational points of view, with the aim of deriving useful evaluation procedures for an intelligent tutoring system such as OFMTutor.

Traditional Instructional Evaluation

As discussed previously, formative and summative evaluation comprise part of the instructional design process. Dick (1977a) describes formative evaluation:

> Formative evaluation may be ... defined as a process of systematically trying out instructional materials with learners in order to gather information and data which will be used to revise the materials. The implication of the term 'formative' is that the evaluation process occurs *while the materials are still being developed.*
>
> ... The sole purpose of formative evaluation is to provide the instructional designer with as much information as possible to revise and strengthen the product which is under development. (pp. 311-312)

The first suggested phase of formative evaluation is one-to-one evaluation. A small representative sample of the target student population (preferably three students -- one of below average ability, one of average ability, and one of above average ability) works through a draft of the instructional materials (including any tests) with the designer. Students give feedback both by their performance and comments to the designer. The designer also asks specific questions in order to discover particular strengths and weaknesses of the materials. This phase is much like a "pilot study" used in traditional experimental situations. The one-to-one phase also includes a review of the materials by a domain expert in order to insure the accuracy of the content. The output from this phase is a set of comments and observations on any difficulties encountered in the use of the materials. The instructional materials are revised with respect to this output, and the revised materials are used in the second phase of formative evaluation.

The second phase of formative evaluation is a small-group evaluation. The purposes of this phase are to evaluate the effectiveness of the first phase's revisions, identify any remaining difficulties, and begin the determination of the feasibility of administering the materials in the field. Dick (1977a) recommends a representative sample of between eight and 24 students for the small-group evaluation. The students take tests and study the materials in a manner similar to that to be used in the field. Questionnaires are also

given, and any helpful comments are solicited as well. The output from the second phase of formative evaluation includes comments, questionnaire data, test scores, and learning times.

The third phase of formative evaluation is field trial evaluation. The major purpose of this phase is to "determine the administrative feasibility of using the instructional materials under normal classroom conditions" (Dick, 1977a, p. 316), as well as to ascertain the effectiveness of the previous revisions. The most critical component of this evaluation phase is that the materials are used in the environment for which they are ultimately intended. Dick (1977a) suggests that the sample size for field trial evaluation should be at least 30 students.

Dick (1977a) notes that no guidelines exist for when to terminate formative evaluation; the decision to terminate formative evaluation "is based almost entirely upon the specific circumstances surrounding the development project" (p. 330). Typically, time requirements or funding are important factors. Designers may also set statistical criteria to be met. For example, the military established an "80/80" criterion for formative evaluation termination. This meant that when 80% of the students achieved 80% of the proposed objectives, the formative evaluation process was judged complete.

In practice, formative evaluation is usually the last stage of the design process. However, very often we are interested in comparing alternative instructional products. Summative evaluation is a process meant to provide data needed this comparison process.

Dick (1977b) reviews several models of summative evaluation. One model, proposed by Gagne and Briggs, has four components: support, aptitude, process, and outcome evaluation. Support evaluation examines the instructional materials, the climate of the teaching environment, parental and peer attitudes, and other factors that may affect learning. Aptitude evaluation, or the evaluation of learner aptitude, is important in that aptitude is significantly correlated with eventual learning outcomes. Furthermore, knowledge of the learner population helps define the generalizability of the summative evalution studies. Process evaluation refers to the documentation of instructional materials and procedures and the formative evaluation process. Outcome evaluation refers to the evaluation and reporting of instructional objectives, criteria for success, and results of product successes and failures.

The basic process of summative evaluation, as described by Dick (1977b), consists of five steps: identification of intended outcomes, identification of the target student population and experimental design, development of the evaluation instruments, documentation of the instructional process, and preparation of the final report. First, one identifies the relevant instructional goals and objectives. Next, one identifies a representative sample of students. Depending on the availability and size of the sample, one has several alternative experimental situations: comparisons between two experimental groups, comparison of one group to norms for a standardized exam, or, in the worst case, to establish criteria for success and administer the appropriate tests to the sample after the completion of their studies. The summative evaluation is also responsible for developing evaluation instruments for data collection. These include means for assessing learning outcomes (e.g., objectives-referenced tests), attitudes about the instructional content and form (e.g., questionnaires), and cost. The instructional process itself must be documented thoroughly, and a clear final report prepared.

Training Program Evaluation

From an industrial-organizational psychology perspective, Landy and Trumbo (1980) review several different approaches to the evalution of training programs. One approach distinguishes between *internal criteria* (i.e., performance in the training situation) and *external criteria* (i.e., performance on the job). Landy and Trumbo give examples of each:

> Internal criteria include objective exams, questionnaires reflecting attitude changes by the trainees, and the opinions of trainees, trainers, or others as to the effectiveness of the program. Comparison of training methods or programs may use the number of hours of training required to reach a common training performance level as an internal measure. A similar criterion -- hours (days or weeks) to reach standard production on the job *after training* -- would be an external criterion. External criteria include measures of quantity or quality of production, time to reach production levels, accident records (for safety training) and other indications of job behavior or training results. (p. 296)

Another approach distinguishes four levels of criteria: reaction, learning, behavioral, and results criteria. The former two correspond to internal criteria, and the latter two correspond to external criteria. Reaction criteria are concerned with the trainees' opinions of the program and typically consist of one or more questionnaires. Learning criteria include final exams, performance tests, and other measures of how much was learned. Learning criteria should reflect the objectives of the training program. Behavioral criteria include performance measures on the job. Results criteria involves the assessment of the utility of training with respect to organizational objectives (e.g., percent increase in job proficiency, percent decrease in accidents or turnover).

Landy and Trumbo also discuss various experimental designs used in the assessment of training programs. Solomon's proposed four group design is one of the most famous and most complete (see Figure 6). The group of prime interest is the experimental group, which is given a pretest, undergoes training, and is then given a posttest. The Control 1 group is given "sham training" during the training period to control for the "Hawthorne effect" (i.e., the effect of *perceived* experimental manipulations on performance; the famed Hawthorne studies showed that workers' productivity increased when they *believed* that working conditions were altered for the better, when in fact the conditions were exactly the same! This phenomena is also known as the "placebo effect"). The Control 2 group is needed to control for the effect of time (i.e., the effect of waiting the duration of the training period before taking the posttest). Finally, the Control 3 group is needed to control for the effect of giving a pretest.

Insert Figure 6

about here

Proposed OFMTutor Evaluation

Formative evaluation is a very important part of any educational project. Because OFMTutor is not intended for simultaneous use by a classroom of students, it is proposed that the one-on-one and small

group evaluations are sufficient for this purpose. The focus of summative evaluation will be on students trained with OFMTutor as compared to those who read over a training manual and operated the system untutored.

Academic research projects typically do not have the opportunity to work with real operators of complex dynamic systems. The OFMTutor evaluation will focus on internal criteria; specifically, reaction (questionnaire) and behavioral (performance) criteria.

REFERENCES

Briggs, L. J. (Ed). (1977a). *Instructional design: Principles and applications.* Englewood Cliffs, NJ: Educational Technology Publications.

Briggs, L. J. (1977b). The teacher as designer. In L. J. Briggs (Ed.), *Instructional design: Principles and applications,* 221-258. Englewood Cliffs, NJ: Educational Technology Publications.

Brown, J. S., Burton, R. R., and de Kleer, J. (1982). Pedagogical, natural language and knowledge engineering techniques in SOPHIE I, II, and III. In D. Sleeman and J. S. Brown (Eds.), *Intelligent tutoring systems,* 227-282. Orlando, FL: Academic Press.

Burton, R. R. (1982). Diagnosing bugs in a simple procedural skill. In D. Sleeman and J. S. Brown (Eds.), *Intelligent tutoring systems,* 157-183. Orlando, FL: Academic Press.

Burton, R. R. and Brown, J. S. (1982). An investigation of computer coaching for informal learning activities. In D. Sleeman and J. S. Brown (Eds.), *Intelligent tutoring systems,* 79-98. Orlando, FL: Academic Press.

Chambers, A. B. and Nagel, D. C. (1985). Pilots of the future: Human or computer? *Communications of the ACM, 28,* 11, 1187-1199.

Clancey, W. J. (1982). Tutoring rules for guiding a case method dialogue. In D. Sleeman and J. S. Brown (Eds.), *Intelligent tutoring systems,* 201-226. Orlando, FL: Academic Press.

Clancey, W. J. (1987). *Knowledge-based tutoring: The GUIDON program.* Cambridge, MA: MIT Press.

Dick, W. (1977a). Formative evaluation. In L. J. Briggs (Ed.), *Instructional design: Principles and applications,* 311-333. Englewood Cliffs, NJ: Educational Technology Publications.

Dick, W. (1977b). Summative evaluation. In L. J. Briggs (Ed.), *Instructional design: Principles and applications,* 337-348. Englewood Cliffs, NJ: Educational Technology Publications.

Fath, J. L. (1987). An architecture for adaptive computer-assisted instruction programs for complex dynamic systems. PhD dissertation, Report 87-3, Center for Man-Machine Systems Research, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA.

Fath, J. L., Mitchell, C. M., and Govindaraj, T. (1988). An ICAI architecture for troubleshooting in complex, dynamic systems. Manuscript in preparation.

Fox, B. A. (1987a). Interaction as a diagnostic resource in tutoring. Dept. of Linguistics and Institute of Cognitive Science, University of Colorado, Boulder, CO.

Fox, B. A. (1987a). Repair as factor in interface design. Dept. of Linguistics and Institute of Cognitive Science, University of Colorado, Boulder, CO.

Genesereth, M. R. (1982). The role of plans in intelligent teaching systems. In D. Sleeman and J. S. Brown (Eds.), *Intelligent tutoring systems,* 137-156. Orlando, FL: Academic Press.

Goldstein, I. P. (1982). The genetic graph: A representation for the evolution of procedural knowledge. In D. Sleeman and J. S. Brown (Eds.), *Intelligent tutoring systems,* 51-77. Orlando, FL: Academic Press.

Govindaraj, T. (1987). Qualitative approximation methodology for modeling and simulation of large dynamic systems: Application to a marine powerplant. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-17*, No. 6, 937-955.

Hollan, J. D., Hutchins, E. L., and Weitzman, L. M. (1987). STEAMER: An interactive, inspectable, simulation-based training system. In G. Kearsley (Ed.), *Artificial intelligence and instruction: Applications and methods*, 113-134. Reading, MA: Addison-Wesley.

Johnson, W. L. and Soloway, E. (1987). PROUST: An automatic debugger for Pascal programs. In G. Kearsley (Ed.), *Artificial intelligence and instruction: Applications and methods*, 47-68. Reading, MA: Addison-Wesley.

Jones, P. M. (1988). Constructing and validating a model-based operator's associate for supervisory control. M. S. Thesis, Report 88-1, Center for Human-Machine Systems Research, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA.

Jones, P. M., Mitchell, C. M. and Rubin, K. S. (1988). Intent inferencing with a model-based operator's associate. Report 88-2, Center for Human-Machine Systems Research, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA. Also to appear in *Proceedings of the Sixth Symposium on Empirical Foundations of Information and Software Sciences*, October 1988, Atlanta, GA.

Kibler, R. J. and Bassett, R. E. (1977). Writing performance objectives. In L. J. Briggs (Ed.), *Instructional design: Principles and applications*, 49-95. Englewood Cliffs, NJ: Educational Technology Publications.

Landy, F. J. and Trumbo, D. A. (1980). *Psychology of work behavior*. Homewood, IL: Dorsey.

Lesgold, A. (1988). Toward a theory of curriculum for use in designing intelligent instructional systems. In H. Mandl and A. Lesgold (Eds.), *Learning issues for intelligent tutoring systems*, 114-137. New York: Springer-Verlag.

Miller, R. A. (1985). A systems approach to modeling discrete control performance. In W. B. Rouse, (Ed.), *Advances in Man-Machine Systems Research*, Vol. 2, 177-248. New York: JAI Press.

Mitchell, C. M. (1987). GT-MSOCC: A domain for research on human-computer interaction and decision aiding in supervisory control systems. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-17*, July/August, 553-572.

Mitchell, C. M. and Miller, R. A. (1986). A discrete control model of operator function: A methodology for information display design. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-16*, 343-357.

Mura, S. S. (1983). Licensing violations: Legitimate violations of Grice's conversational principle. In R. T. Craig and K. Tracy (Eds.), *Conversational coherence: Form, structure, and strategy*, 101-115. Beverly Hills, CA: Sage Publications.

Nii, H. P., Feigenbaum, E. A., Anton, J. J., and Rockmore, A. J. (1982). Signal-to-symbol transformation: HASP/SIAP case study. Heuristic Programming Project, Report No. HPP-82-6, Heuristic Programming Project, Stanford University, Stanford, CA.

Nii, H. P. (1986). Blackboard systems. *AI Magazine, 7-2, 7-3*.

Norman, D. A. and Draper, S. W. (Eds.) (1986). *User centered system design: New perspectives on human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Novak, J. D. (1986). *A theory of education.* Ithaca, NY: Cornell University Press.

Park, O., Perez, R. S., and Seidel, R. J. (1987). Intelligent CAI: Old wine in new bottles, or a new vintage? In G. Kearsley (Ed.), *Artificial intelligence and instruction: Applications and methods,* 11-45. Reading, MA: Addison-Wesley.

Rasmussen, J. (1986). *Information processing and human-machine interaction: An approach to cognitive engineering.* New York: North-Holland.

Richer, M. H. and Clancey, W. J. (1985). GUIDON-WATCH: A graphic interface for viewing a knowledge-based system. *IEEE Transactions on Computer Graphics and Applications,* November, 51-64.

Rubin, K. S., Jones, P. M. and Mitchell, C. M. (1987). OFMspert: Application of a blackboard architecture to infer operator intentions in real time decision making. Report No. 87-6, Center for Human-Machine Systems Research, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA. Also *IEEE Transactions on Systems, Man, and Cybernetics,* to appear.

Schank, R. C. and Abelson, R. P. (1977). *Scripts plans goals and understanding.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Sheridan, T. B. and Johannsen, G. (Eds.) (1976). *Monitoring behavior and supervisory control.* New York: Plenum.

Towne, D. M., Munro, A., Pizzini, Q. A., Surmon, D. S., and Wogulis, J. (1988). ONR final report: Intelligent maintenance training technology. Technical Report No. 110, Behavioral Technology Laboratories, Department of Psychology, University of Southern California, Redondo Beach, CA.

Wenger, E. (1987). *Artificial intelligence and tutoring systems: Computational and cognitive approaches to the communication of knowledge.* Los Altos, CA: Morgan Kaufmann.

Wickens, C. D. (1984). *Engineering psychology and human performance.* Columbus, OH: Charles Merrill.

Woods, D. D., (1986). Cognitive technologies: The design of joint human-machine cognitive systems. *The AI Magazine,* 86-92.

Woolf, B. P. (1986). Teaching a complex industrial process. COINS Technical Report 86-24, Computer and Information Science, University of Massachusetts, Amherst, MA.

Woolf, B. P. (1987). Theoretical frontiers in building a machine tutor. In G. Kearsley (Ed.), *Artificial intelligence and instruction: Applications and methods,* 229-267. Reading, MA: Addison-Wesley.

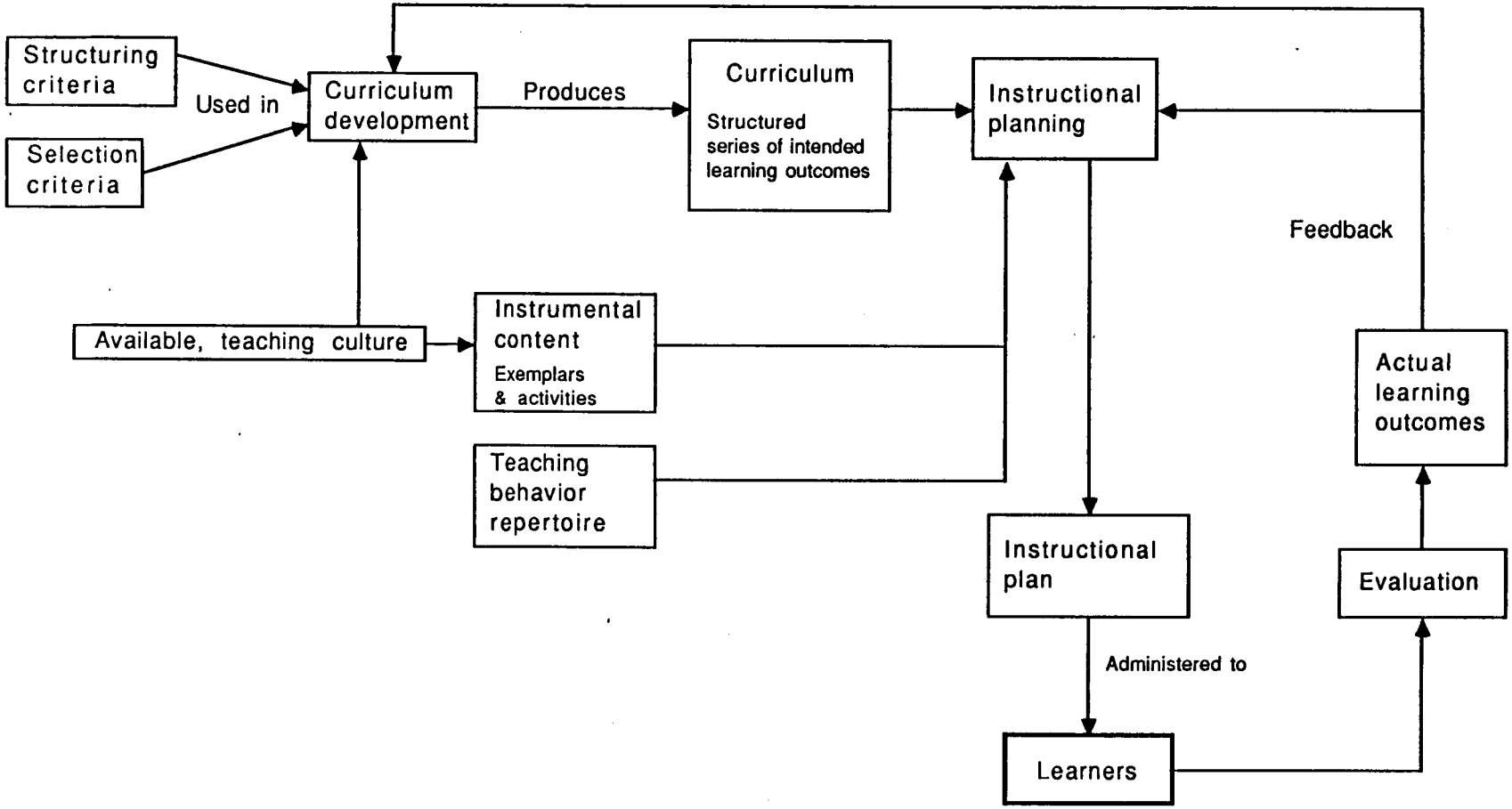

Figure 1. A simplified version of Johnson's model of curriculum. Adapted from Novak, 1986, p. 132.

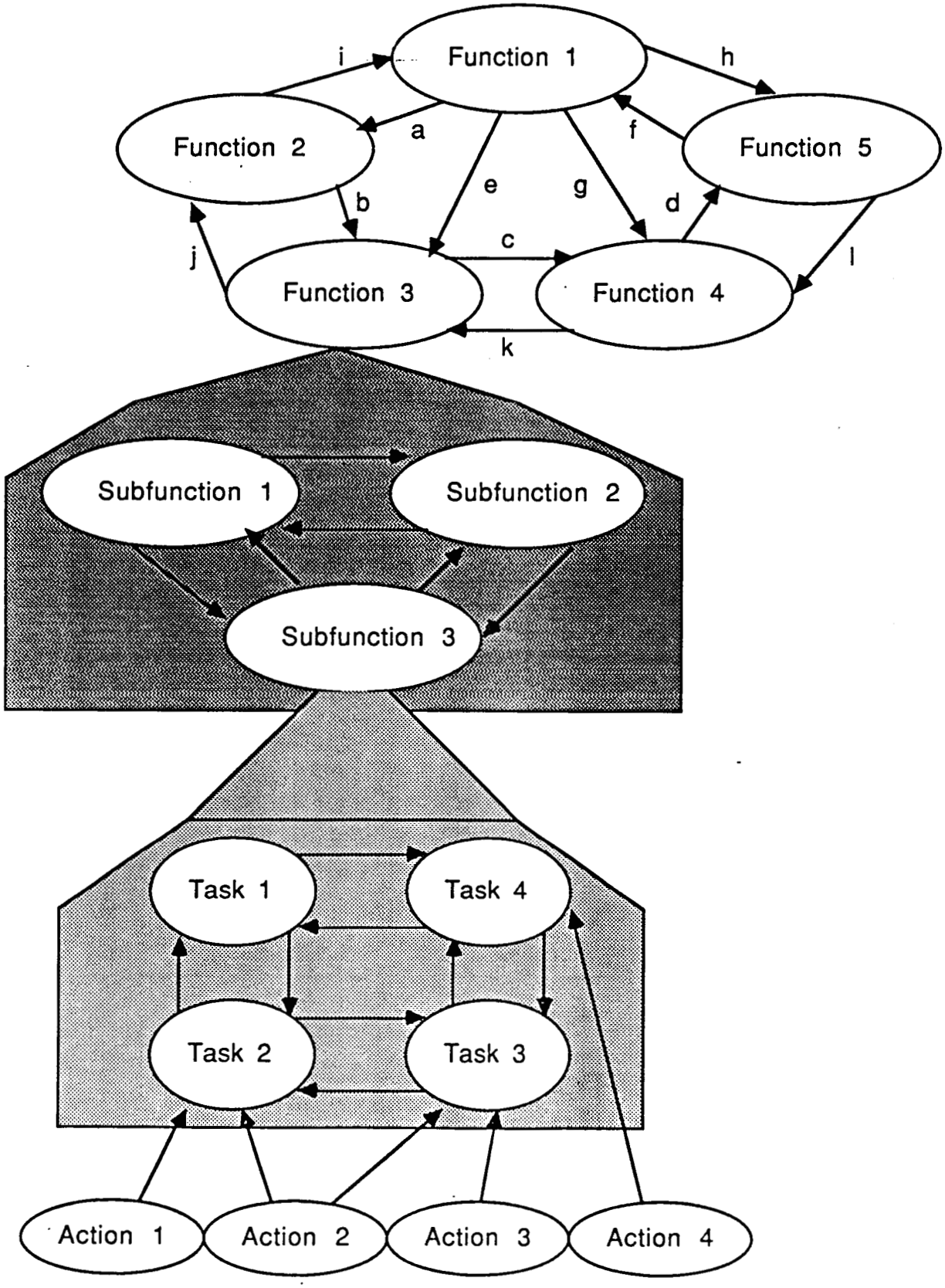

Figure 2. A Generic Operator Function Model

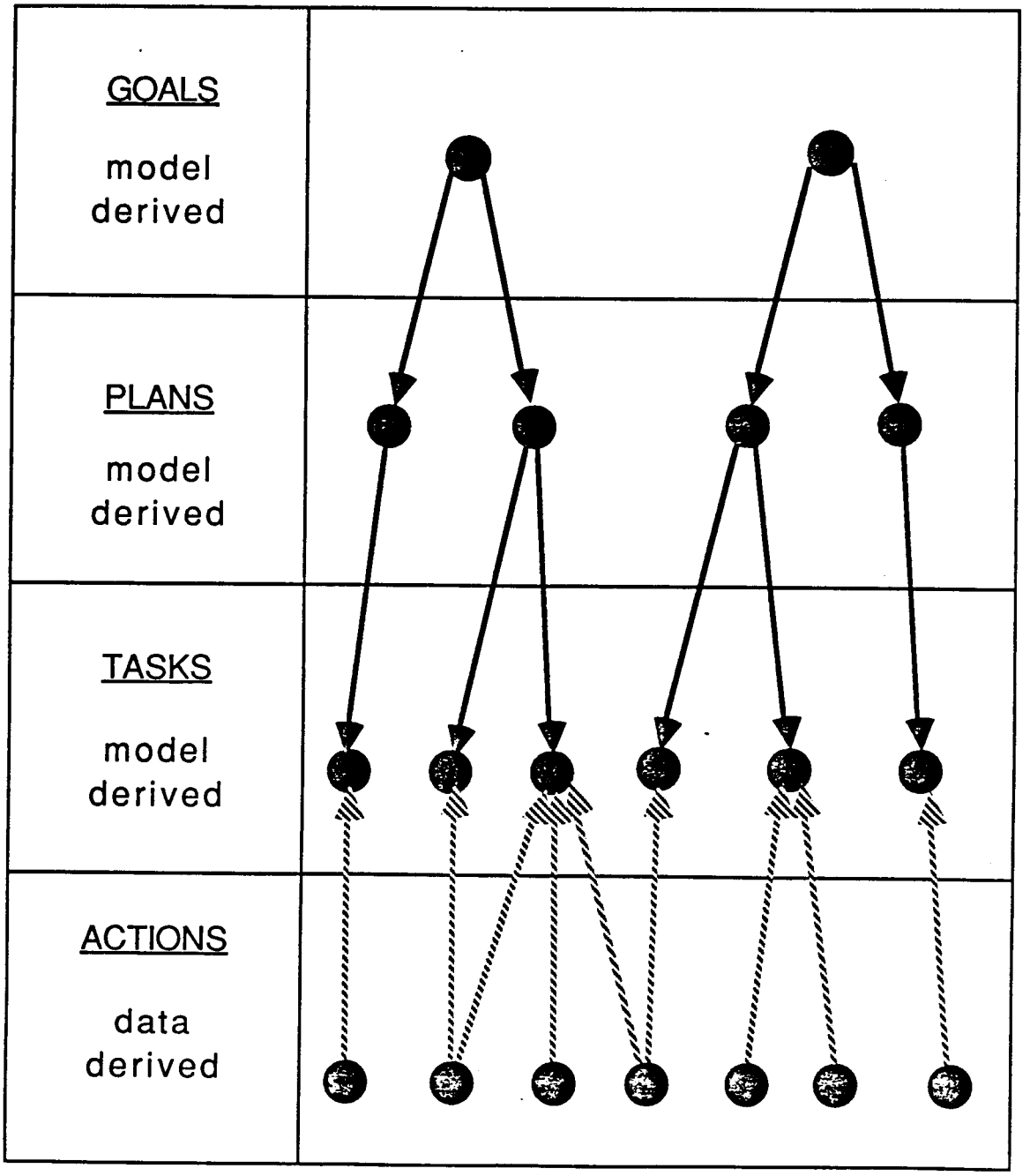

Figure 3. The Blackboard Intent Inferencing Structure

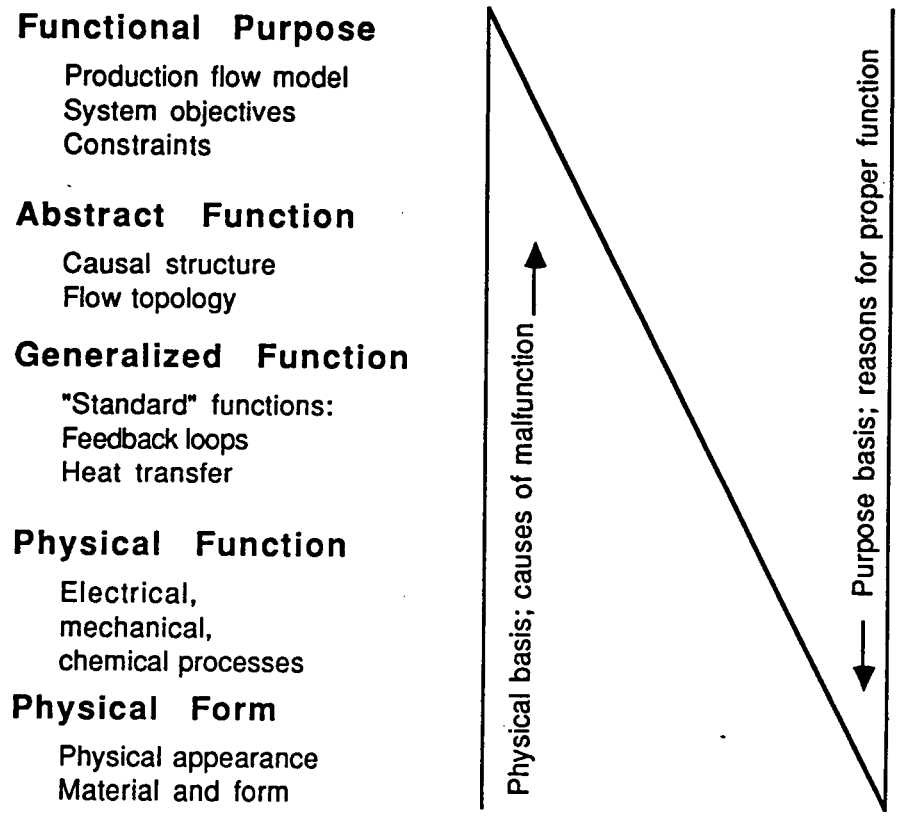

Figure 4. The abstraction hierarchy. Adapted from Rasmussen, 1986, p. 15.

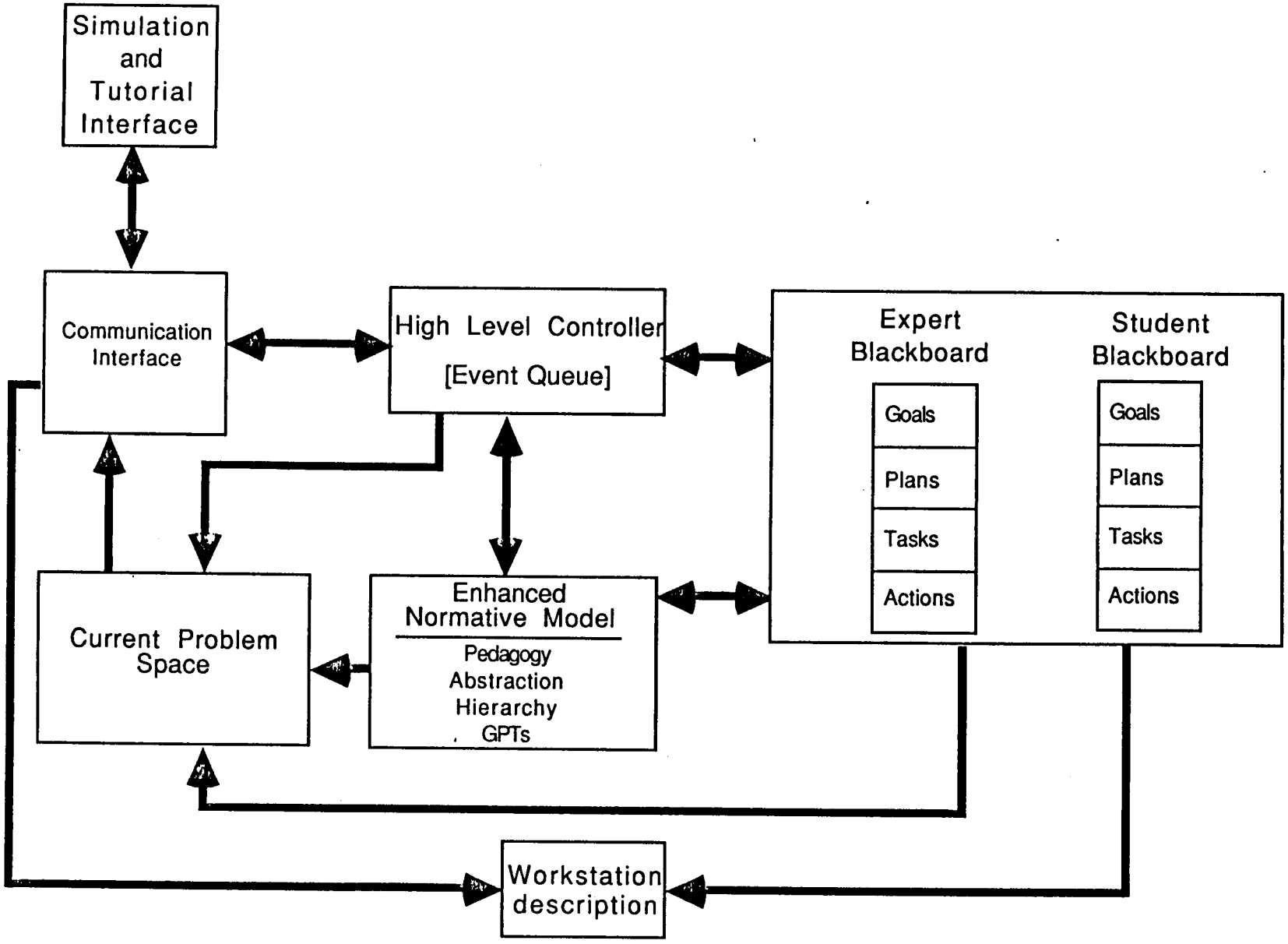

Figure 5. OFMTutor Architecture

Group	Before	Training Period	After
Experimental	Test	Train	Test
Control 1	Test	Placebo activity	Test
Control 2	Test	No train	Test
Control 3	No test	No train	Test

Figure 6. Four-group experimental design proposed by Solomon. From Landy and Trumbo, 1980, p. 300.

Table 1. Taxonomy of capabilities and actions.
Adapted from Briggs, 1977, p. 69.

Capability	Action	Example
Intellectual Skill		
Discriminates	Discriminates	Distinguishes sounds
Concrete Concept	Identifies	Names computer components
Defined Concept	Classifies	Classifies using definition
Rule	Demonstrates	Solves linear equations
Problem Solving	Generates	Synthesizes rules to generate solution
Cognitive Strategy	Originates	Applies model of diffusion to originate solution to reduction of air pollution
Information	States	States current events
Motor Skill	Executes	Drives a car
Attitude	Chooses	Chooses to share toys

Table 2. Suggested guidance for particular learning outcomes. Adapted from Gagne, 1977, p. 211.

Learning Outcome	Suggested Guidance
Discrimination	Point out distinctive features of objects to be discriminated
Concrete Concepts	Gives cues to identifying attributes
Defined Concepts	Present component concepts in proper sequence
Rules	Show how component concepts make up the rule
Problem Solving	Provide minimum cues needed to select and apply rules
Cognitive Strategies	Provide only indirect cues
Names and Labels	Provide coes or memory bridges
Facts	Provide meaningful context
Organized Knowledge	Provide prompting in the context of the organizational framework
Motor Skills	Stimulate recall of sequence of acts; provide practice with feedback
Attitudes	Show human model behavior and how reinforced

Table 3. Some principles of interaction for a coach. Adapted from Burton and Brown, 1982.

Principle 1: Before giving advice, be sure that the issue is one in which the student is weak.

Principle 2: When illustrating an issue, use an example (alternative action) that is dramatically superior to the action taken by the student.

Principle 3: After giving the student advice, allow an opportunity for redoing the action.

Principle 4: If a student is close to making a serious mistake, interrupt and tutor only with advice that will prevent that mistake.

Principle 5: Do not tutor on two consecutive actions.

Principle 6: Allow the student to explore before tutoring.

Principle 7: Praise the student when appropriate.

...

Principle 10: If a student asks for help, provide several levels of hints.

...

Principle 12: Be forgiving of possibly careless errors.

Table 4. Fox's suggestions for dialogue management. Adapted from Fox, 1987b, p. 12.

Turn-taking should not be an on-off option. The interface must allow for each party to participate as they see fit.
It is especially important that during a turn, the other party have the ability to show understanding, initiate repair, etc., at the end of every conversational unit.
The turn-taking mechanism must provide flexibility in turn length.
Correction of the student, or initiation of such correction, should be withheld until the student has had an opportunity to self-correct, or initiate self-correction.

Table 5. Woolf's adaptions of Gricean maxims for discourse. Adapted from Woolf, 1987, p. 254.

Quality	Be committed and interested in student's knowledge. Be supportive and cooperative. Do not take the role of "antagonist"
Quantity	Be specific and concise. Use a minimum of attributes to describe a known concept.
Relation	Be relevant. Find the student's threshold of knowledge. Bring up new topics and viewpoints as appropriate.
Manner	Be in control. Allow both the student and the context to determine the topic.

Table 6. Tutoring maxims supported by conversational move-classes. From Woolf, 1987, p. 255.

Be Cooperative	
Work with student	Explain, summarize, review or repeat, and clearly terminate topics. Release control of dialogue.
Be Committed	
Show interest	Acknowledge answer. Explain topics.
Support student	Outline, introduce topics.
Be Relevant	
Find student's threshold	Question student. Evaluate student hypotheses. Propose and verify misconceptions.
Teach at threshold	Provide analogy examples. Summarize topic.
Be Organized	
Structure domain	Outline, introduce, terminate, review topics.
Complete information	Clearly terminate topics. Teach subtopics and attributes after topic. Teach subgoals after goal.
Be In Control	
Strictly guide discoure	Introduce, describe topic. Question student.